UNIVERSITY OF CALIFORNIA,
IRVINE

Model Performance as an Estimator of Language Complexity

THESIS

submitted in partial satisfaction of the requirements
for the degree of

MASTER OF SCIENCE

in Computer Science

by

Peter William Schaedler

Thesis Committee:
Assistant Professor Sameer Singh, Chair
Assistant Professor Richard Futrell
Assistant Professor Stephan Mandt

2019

TABLE OF CONTENTS

LIST OF FIGURES

Page

LIST OF TABLES

Page

ACKNOWLEDGMENTS

I would like to thank Professors Sameer Singh, Richard Futrell, and Stephan Mandt for their guidance, suggestions, and encouragement.

I would also like to thank my friends and family for supporting me and my goals, even when I may be far away.

Finally, I would like to thank Jodaiko and the collegiate taiko community for keeping me sane over the past two years.

ABSTRACT OF THE THESIS

Model Performance as an Estimator of Language Complexity

By

Peter William Schaedler

Master of Science in Computer Science

University of California, Irvine, 2019

Assistant Professor Sameer Singh, Chair

Quantifying the complexity of a natural language is a difficult task on its own and comparing two or more languages typically requires establishing a reference point and determining the biases and context of the languages being compared. I propose a new metric for unbiasedly quantifying the complexity of a language in a way that allows for easy comparison between languages. I use a variety of common machine learning solutions for tasks such as part-of-speech tagging and language modeling, then analyze the learning ability of these models as parameters are adjusted. I then use the evaluation metrics from these tasks to compare similar models trained on different languages. I find that the evaluation metrics accuracy and perplexity mimic the behavior of four metrics found in linguistics literature and can be used to compare relative complexities.

Chapter 1

Introduction

Quantifying the complexity of a language is an open problem in linguistics, and there is no accepted method for measuring and comparing complexities [3, 13]. First, "complexity" needs to be defined more specifically. Different forms of language complexity have been researched, including morphological complexity, syntactic complexity, size of vocabulary, and others. These different types of complexities each account for a certain component of the language, but a single metric to quantify the complexity of a language has yet to be established. Defining one single quantity to cover all of the various components of a language is not only difficult, but can lead to misguided conclusions when comparing two or more languages. For example, one language may have a comparatively simple grammatical structure but many different words in its vocabulary, while another could have a very rich grammar which requires fewer unique words.

When comparing different languages in a research context, individual components are usually compared independently of each other. However, when people casually describe the differences between languages, they often view it holistically. In addition, casual comparisons of languages are often held with respect to a given language, for example comparing

the language being used to some other language. Native speakers of a language may find some languages more difficult than others to learn or understand. Languages that come from a very different language family or use a different writing system for written language may be far more difficult than a language from the same family and that uses the same alphabet or character set. This inherently introduces a bias into comparisons of languages, as they are being viewed from the reference point of a given language.

Machine learning algorithms have allowed computers to find patterns in data sets that previous statistical methods were not able to uncover. Furthermore, new methods have allowed large data sets to be used to train these algorithms for a variety of tasks, including vision and image recognition, financial data analysis, and even language recognition and generation. Advances in neural network algorithms especially have allowed the field of natural language processing to proliferate. Even without a background in computer science, statistics, or linguistics, a hobbyist can bring together tools and a sufficiently large data set and build a language model to perform tasks such as part-of-speech tagging, language generation, sentiment analysis, and summarization.

Because these models learn only from the data they are given, they lack the inherent bias that humans have comparing different languages. By observing how computers learn various languages, I aim to determine a more unbiased metric of how complex languages are relative to each other. This would account for various writing systems as well, given sufficient text encodings for each system.

In this thesis, I propose a new metric for quantifying the complexity of languages. I train a variety of machine learning models for both part-of-speech tagging and language modeling and generation tasks for four sample languages: English, Spanish, Japanese, and Chinese. These languages come from different language families and use three different writing systems. English and Spanish use a shared writing system with the Latin alphabet, and also share similarities in their language due to historical influence. Japanese and Chinese are very

different from English and Spanish. While Japanese and Chinese do use different writing systems, the Japanese system does make use of some Chinese characters. In addition, I have personal experience with English, Spanish, and Japanese that I hope will help with analysis and interpreting my results.

I run experiments using the models, varying the parameters used for training and the amount of data used to train the models, in order to understand how quickly each model learns its language. By using common accuracy metrics for testing data sets for each model and comparing the model's performance over the different tests, I establish a score for how difficult the language is for the model to learn. These scores can then be compared between the different languages to understand the level of complexity for a given language. For rigor, I also compare the calculated scores to four scores calculated from methods used in linguistics literature to determine the morphological complexity of a corpus.

While these tests focus on complexity of written language, this method could be generalized to spoken language as well, given a method of retrieving and quantifying audio data for input into the models, along with an evaluation metric. I focus on written language because of the abundance of text data available without the need for training labels of correct outputs. That being said, labeled data is used for part-of-speech classification tasks.

To analyze the relationships between the three languages, I use two main tasks: part-of-speech tagging using a logistic regression classifier and language modeling using LSTM-based neural networks. These provide two varying levels of complexity of the system used for the task, going from a very simple classifier to state of the art neural network architectures. For the part-of-speech tagging task, I only use English, Spanish, and Japanese data due to lack of a common dataset source, but for language modeling I use all four languages.

The aim of this thesis is to model morphological complexity specifically, as morphology focuses on the structure of words in language and I hope using word data with machine

learning models will capture the essence of each language's morphology. However, because the data sets used do not include more detailed information on the component morphemes that make up each word, the experiments do not strictly capture morphological complexity. Further discussion on what precisely the experiments do measure is in chapter 5, but through the rest of the thesis I will refer to the goal as morphological complexity.

Overall, the goal of this new metric is to provide an easily-calculable method for comparing the complexity of two corpora that does not rely on linguistic knowledge or a baseline reference language. Because the metric will be calculable from common evaluation metrics, a given user will be able to create a machine learning model, train it using each corpus with minimal preprocessing, and calculate the metrics necessary. From there, comparison between the two corpora is simple and straightforward. Depending on the size of the corpus, the models should give similar results to the experimental results shown in this thesis. This allows for easy reproducibility between corpora.

Researchers studying complexity could also use this easy to calculate metric as a simple and reliable baseline when developing other, more rigorous metrics for complexity that can be used to compare languages. The knowledge that this metric brings about the differences in complexities between languages can also be useful when studying other similarities and differences between languages, or can be used when studying low resource languages, or languages without much available text for analysis. Because this method does not rely on understanding a language, low resource languages or ancient languages can be analyzed just as easily given the necessary encodings for their writing systems.

Chapter 2

Background and Related Work

This thesis draws on knowledge from two domains: morphological complexity research in linguistics literature and machine learning and language modeling research in computer science literature.

2.1 Morphological Complexity of Natural Languages

Language complexity has many different subfields of research including morphological complexity, grammatical complexity, phonetic complexity, syntactic complexity, and more. However, in general languages are believed to all be similarly complex [20]. However, languages can be more or less complex in certain aspects. Morphological complexity refers to the complexity of words in a language, and how they connect to other words. Bane [2] notes that morphological complexity is a good domain to explore complexity metrics because of its inherent connection to other forms of linguistic complexity and because it seems clear that some languages are more morphologically complex than others.

Various measures of language complexity have been proposed, but there appears to be no standard for measuring complexity of language. Research from conferences such as the Measuring Language Complexity (MLC) workshop contributes to the growing number of metrics that can be used. From that workshop, Ehret [9] proposes Kolmogorov complexity (discussed in further detail in this chapter) as a universal measure of language complexity, and von Prince and Demberg [28] examine perplexity of part-of-speech tagging tasks as a measure of syntactic complexity. This thesis examines Kolmogorov complexity as a reference metric, and POS tagging is used in my experiments, but I measure accuracy using a simple logistic regression classifier rather than perplexity of n-gram models.

Many metrics for measuring morphological complexity can be separated into two main sets: those that are based on information theory and those that are not. More detail on each set is given below. The work in this thesis likely falls into the information-theoretic group, because our measure is based on the abilities of computers to parse information and react accordingly. Many algorithms have been proposed for calculating measures of complexity, but using machine learning algorithms is a mostly unexplored area. However, machine learning algorithms have been used to observe language generation tasks and see if they follow the same statistical laws as natural language, mirroring their inherent complexity [26].

2.1.1 Non-Information Theoretic Measures

The first set of morphological complexity metrics is those based on counting approaches. A simple example of this is counting words in a corpus, where a corpus with more unique words may be considered more complex. Other examples include counting terms for colors, what Bentz et al. [3] refer to as "Type/Token Ratio," which is the ratio of unique words over total words in a corpus, and other calculated metrics or constants such as Yule's K, Zipf's Z, and Golcher's VM [14]. Even more metrics can be calculated by considering different

morphemes within words themselves, such as prefixes and suffixes. An example metric would be counting unique prefixes in a language.

There are other counting-based approaches to complexity that consider the broader sentence structure. One example structure that can be used is a sentence's syntax tree, or a tree of each word's dependence on another word in a clause. Liu [18] presented mean dependency distance, or the average number of words between a word and its dependent in a sentence as a metric of complexity for language comprehension. This relationship between words is more important in oral language where a listener may have to remember context words of a clause before the main idea is presented.

2.1.2 Information Theoretic Measures

The other main type of metric is information-theoretic metrics. These approaches are based on Shannon's concept of entropy and information theory [25]. Entropy is defined as

$$H = -\sum_{i} P_i \log P_i.$$

Shannon's entropy in the context of language describes the average information content of words where the probabilities of the equation refer to the probability of words appearing in a given vocabulary. Bentz [3] notes that using frequencies normalized by the number of total tokens in a corpus will underestimate true entropy because there will be unseen tokens in a corpus compared to the whole language. However, entropy is relatively straightforward to approximate, making it a common metric of complexity for a corpus [3, 13, 14, 24].

Another related metric is Kolmogorov complexity [16], which measures the amount of information in a sentence by finding the most efficient representation of that sentence. Shannon's

entropy is an upper bound of Kolmogorov complexity up to a constant [13]. It is more difficult to approximate Kolmogorov complexity, but a common approximation focuses on the idea of a description length of a string, or the length of an efficient description of the string [2, 24]. One approach to finding an efficient representation of a given string is to apply a lossless data compression algorithm to it to reduce its size as encoded by a computer [2, 9].

As with counting measures, information theoretic metrics can also apply when considering different morphemes of words. Bane [2] uses the description lengths of affixes, stems, and "signatures" of words in a corpus to calculate his own morphological complexity metric as an upper bound of Kolmogorov complexity.

2.2 Machine Learning

Machine learning is a branch of artificial intelligence research focused on defining statistical models typically to predict some outcome or label given some input data. The input data is in the form of various features or characteristics that define each data point. Various machine learning algorithms exist for both classification and regression problems. Some examples include linear and logistic regression, decision trees, support vector machines, and artificial neural networks.

2.2.1 Logistic Regression Classifiers

Logistic regression classifiers output probabilities by learning parameters for a logistic function of the form

$$f(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1}.$$

Because there are almost no hyper-parameters (or parameters the model designer can choose that are not learned) for this model aside from adding regularization, logistic regression can act as a very simple classifier. One use for this classifier is in part-of-speech tagging, which is a multi-class classification problem [22].

2.2.2 Artificial Neural Networks

A more complicated set of algorithms for machine learning is artificial neural networks. At their most basic, neural networks are connected sets of perceptron units. Perceptrons, designed to model how the human brain learns information, consist of any number of inputs added together as linear combination with weights with an activation function applied to the result [23]. Neural network architectures have the benefit of being very scalable to have many units per layer as well as many layers. Because of this flexibility, they can also learn to model very complex functions efficiently.

Mikolov et al. [19] found that recurrent neural networks (RNNs) work particularly well for the tasks of language modeling and speech recognition. RNNs make use of sequential data, such as time-series to learn to predict elements of the sequence. Text data, either in the form of words or even individual characters, can be seen as sequential data, making RNNs an obvious choice.

However, one drawback of normal RNNs is that in a long sequence, information from the beginning of the sequence loses its weight when predicting towards the end of the sequence. Hochreiter and Schmidhuber [12] developed the Long Short-Term Memory architecture (LSTM) to address this problem. LSTM units allow relevant information in the hidden state to continue passing through the sequence as the network learns and blocks information that is less important. As of writing, LSTM units combined with other techniques still provide state-of-the-art performance for language modeling tasks [10].

Other architectures, including convolutional neural networks, are also used for language modeling to work around the shortcomings of RNNs [6]. Another development is the use of Attention mechanisms and Transformer architectures, developed by Vaswani et al. [27]. Transformer architectures consist of an encoder to transform the sequence of words into a sequence of continuous representations, and then a decoder to transform that back to sequences of words or symbols. For machine translation tasks, Transformer architectures have achieved state-of-the-art results [8].

Using Transformers, Devlin et al. [7] developed BERT, which allows for pre-trained Transformer models to be easily applied to a variety of tasks by adding a single additional domain-specific layer. BERT, which stands for Bidirectional Encoder Representations from Transformers, uses Transformers by passing over input data from both directions. With very large datasets and computational resources, BERT models are able to produce close to state-of-the-art results for a variety of domains with an easily fine-tunable output layer.

2.3 Language Modeling Theory

Aside from studying how to model language, it is important ask what language models are able to capture. Takahashi and Tanaka-Ishii [26] studied whether neural networks are able to learn statistical laws behind natural language, finding that in many cases they are. They found that LSTM language models will produce output that matches statistical laws found in natural language, particularly Zipf's law and Heaps' law. However, correlation began to decrease as sequence lengths increased.

Finally, Cotterell et al. [5] looked at if all languages are equally hard to language model. This is a similar topic to the content of this thesis, but from a different perspective. They found that differences in morphology do contribute to differences in language modeling difficulty,

leading to the conclusion that not all languages are modeled the same and therefore have differing morphological complexities. They developed different language models for each language that perform equally well with a focus on Indo-European languages. In this thesis, I focus on developing the same language models but training on different languages to see how the models behave given different language data.

Chapter 3

Methods

I look at and compare four languages: English, Spanish, Japanese, and Chinese. For each language, I consider two different tasks and then compare the results between languages for similarly constructed models. The two tasks are part-of-speech tagging using a logistic regression classifier and language modeling using an LSTM-based neural network.

For all experiments and processing, I used Python 3 for the programming language along with libraries for scientific computing and machine learning, detailed below.

3.1 Data and Languages

I look at the four languages mentioned for a few reasons. First, many similar studies written in English have focused mainly on Indo-European languages, with not much research comparing to Asian languages such as Japanese and Chinese. Part of the reason for this may also be because of the different writing systems used for Chinese and Japanese. However, because we are able to encode Chinese and Japanese writing just as easily as English writing, machine learning models will have no trouble learning one compared to the other on the

basis of writing system alone. Second, the four languages come from three different language families, providing a variety of backgrounds and origins. Third, it may be interesting to note that English has been influenced quite heavily by Spanish, and Japanese has been influenced especially by Chinese and also by English. These relationships may have some effect on the results. Last, I have experience speaking and reading English, Spanish, and Japanese, and I hope that experience will allow for more in-depth analysis.

The data for all of the experiments comes from the Open Subtitles 2016 data set [17]. This is a large data set of labeled and unlabeled movie subtitles. It is also a parallel corpus, meaning the different languages are translations of the same original sources. This is particularly helpful for comparing results between languages.

For English, Spanish, and Japanese, the Open Subtitles data set provides labeled data, including part-of-speech data for each word. However, there is no labeled data for Chinese. Therefore, for the part-of-speech tagging task, I will not be including Chinese and instead focus on English, Spanish, and Japanese. The data set also includes various other labels, including word stems, dependency data, and alignment data to other languages.

For the language modeling task, I make use exclusively of the word data provided. The dataset includes words pre-tokenized and separated, so collecting a list of individual raw tokens is trivial.

The dataset for each language is separated into multiple years, subfolders, and then individual XML files within each subfolder containing the data separated by sentence and word with included tokens for beginning and end of sentence. For the sake of memory space and processing time, I took a subset of these folders, read through all of the XML files within them and combined all of the word data into a single data frame containing its useful metadata. For the part-of-speech tagging, this meant also taking part-of-speech data. For language modeling, this meant taking only a list of tokens.

Table 3.1: Results for each of the reference metrics for English for LSTM model data sizes.

Data Size	Vocab Size	T/T Ratio	Bane Ratio	Kolmogorov
2,500	2673	0.1551	0.08359	0.8388
3,750	3328	0.1283	0.07350	0.7134
5,000	3822	0.1111	0.06876	0.6551

3.1.1 Preprocessing

In order to have as simple and straightforward pipeline as possible, I perform no preprocessing on the raw tokens gathered from the dataset. Punctuation is left in the dataset, typically as their own tokens. Capitalizations are likewise left unchanged. This potentially causes an inflated list of unique tokens. However, it provides a more even comparison to languages like Japanese and Chinese that have no concept of capitalization.

Additionally, I do not perform any explicit feature extraction. The input of each model is just the words themselves without any other supplemental data. While this may prevent the models from performing optimally, again it allows for a more balanced comparison between languages that may not share similar features.

3.2 Reference Metrics

To ensure consistency with established metrics from linguistics literature, I compare my metric to four other metrics. Two of these are non-information theoretic metrics, while the other two are information theoretic. Each metric was calculated using a subset of the corpus the same size as the training data for each task. Each instance was calculated five times and averaged to account for randomness in sampling the training data. The results for English can be found in Table 3.1. The full results for all four languages can be found in the Appendix.

3.2.1 Vocabulary Size

The first, and simplest, is simply the number of unique tokens found in the corpus. Perhaps the simplest possible baseline, this metric gives a rough idea of the number of types found in the language. More types means more morphological complexity.

3.2.2 Type/Token Ratio

The second is type/token ratio as defined by both Bentz et al. [3] and Gutierrez-Vasques and Mijangos [11]. This is defined as the ratio of unique words to the total number of tokens in the corpus. Following convention from Bentz et al.,

$$C_{TTR} = \frac{V}{\sum_{i=1}^{V} fr_i}$$

where V is the number of unique tokens and fr_i is the frequency of the i^{th} token. This gives a score between 0 and 1 which is higher when there is a greater number of unique tokens and fewer repeated tokens.

3.2.3 Bane Ratio

The third metric is one proposed by Bane [2]. In his paper, he refers to it solely as "morphological complexity," so I refer to it here as the Bane Ratio. He defines it as

$$\text{Morphological Complexity} = \frac{DL(\text{Affixes}) + DL(\text{Signatures})}{DL(\text{Affixes}) + DL(\text{Signatures}) + DL(\text{Stems})}$$

where $DL(x)$ is the description length of x. The description length is computed as an approximation to Kolmogorov complexity, making this an information theoretic metric. To implement this metric, I use Bane's research tool Linguistica [15] to parse the affixes, signatures, and stems from the words in the corpus and then compute description length of the lengths of those strings.

Because Linguistica assumes a Latin alphabet for its calculations, I do not calculate Bane ratio for Japanese and Chinese.

3.2.4 Kolmogorov Complexity

Finally, the last metric is Kolmogorov complexity. I approximate this using the method used by Bane [2] and Juola [13]. I join the entire corpus into one string separated by spaces, convert them to a byte string in Python, then compress the string over 1,000 iterations using gzip. Finally, to convert the resulting compressed string into a numerical score, I compute the ratio of the size of the compressed string to the original, giving a complexity score between 0 and 1. In the case that the resulting string is larger than the original, I take the minimum of the resulting score and 1.

$$K = \min\left(\frac{\text{len}(compressed)}{\text{len}(original)}, 1\right)$$

The act of compressing each corpus 1,000 times actually causes its complexity score to increase, as seen in Figure 3.1. All languages showed the same linear growth after an immediate drop off from the first compression. However, to maintain consistency with the method proposed by Bane and Juola, I keep the method as is.

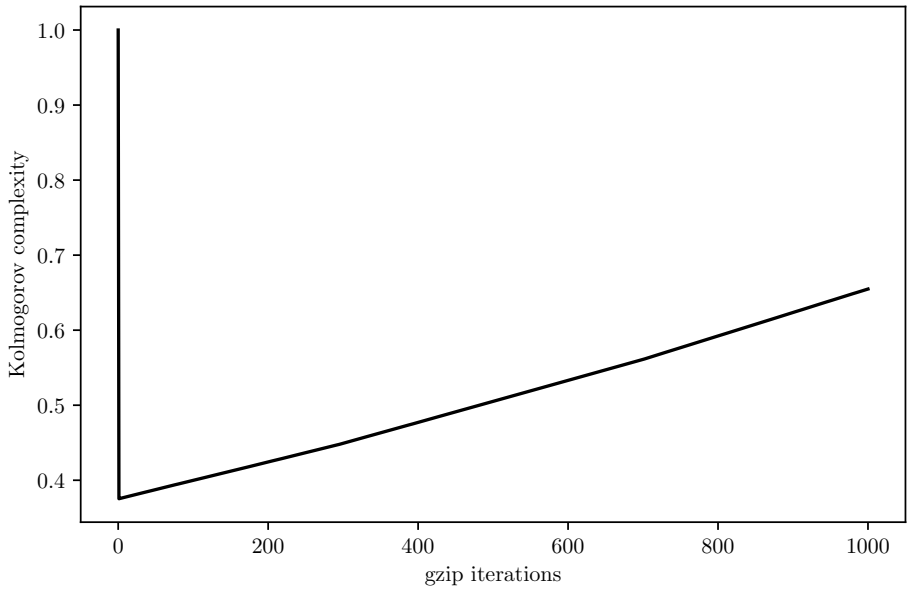

Figure 3.1: Example growth of English Kolmogorov complexity vs gzip iterations.

3.3 Experiment Tasks

There are two experiments I run to analyze the way each language is learned.

3.3.1 Part-of-Speech Tagging

The first is part-of-speech tagging using a logistic regression classifier. This task is to serve as a very simple baseline of what a machine learning algorithm can learn about a language with a very rigidly structured problem. In this case, the model is given an input word or token and must determine its part of speech. Because this is simply a multi-class classification problem, I can solve this with a simple logistic regression classifier.

The model is trained on a sequence of words with their corresponding part-of-speech labeled. Because there are no additional features with the input, each word acts as a unigram with no additional context. The model outputs a probability for each part of speech for what

that word is most likely to be, and the class with the highest resulting probability is chosen as the predicted class.

The logistic regression classifier is implemented using the standard LogisticRegression class from scikit-learn [21] set for multi-class classification and using the lbfgs optimizing algorithm.

For this task, there are only two parameters I can vary: the size of the training data, and the amount of regularization. In this task I apply L2 regularization. I train the classifier with training sizes of 50k, 100k, 250k, 500k, and 1 million tokens. This is after a 75/25 percent training/testing split of the data, so I would have a testing set of 16.6k, 33.3k, etc. For the regularization, I use values 1.0, 0.99, 0.975, 0.95, and 0.9, where smaller values correspond to stronger regularization. I perform tests for all combinations of values, meaning 25 tests per language for three languages.

The metric used to evaluate this task is the accuracy of predictions, i.e. the percentage of correct predictions over the testing set.

3.3.2 LSTM Language Modeling

The task of word-level language modeling in natural language processing involves creating a statistical model of language that, given a sample text, can predict the next most likely word that follows. For example, given the text "After chapter one comes chapter" a good language model might predict the most probable word to be "two." A bad model might predict "dog." Of course, a model may also predict "three," which is also possible but not the best answer in this context.

The second task involves creating a language model using LSTM units. The models are implemented using the Keras [4] library for Python with a TensorFlow [1] backend. The

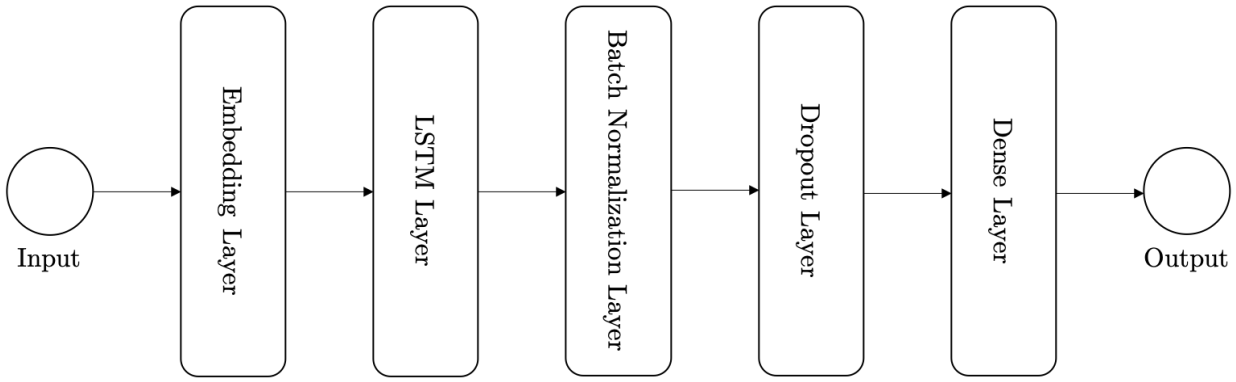

Figure 3.2: LSTM model architecture.

model architecture for the LSTM language models can be seen in Figure 3.2. Input is sent to an embedding layer than learns a spatial representation as a continuous value for each of the words. Words are given close values if their usage and meaning if learned to be similar. This adds a feature layer to the input. Next they are sent to an LSTM layer with a variable number of units. After they go through both a batch normalization layer and a dropout layer. This is to combat overfitting the training data, which became a common issue throughout the learning process. Finally the data is put through a dense layer with a softmax activation to produce prediction probabilities.

The input is given as a vector of input sequences. Each sequence represents a full sentence from the input data. The raw sentences, split by word, are tokenized and converted to a vector of integers where each number is between 1 and the number of unique tokens. Then each sequence is padded to the left so that every sequence is the same length. The sequences are generated by taking individual sentences and generating sub-sequences of various lengths, with one word after held out for an output label. The outputs, when fed to the model for training, are converted to a one-hot vector format where the vectors are the length of the total number of unique tokens. This corresponds to what the output of the model is after the dense layer, a vector of probabilities for each possible token. The largest probability is chosen as the predicted token.

There are a number of parameters that can be adjusted for the LSTM models. For these experiments, I will vary size of the training set and number of LSTM units. With the four languages I observe, this gives us 36 different configurations to analyze. Training size is observed at 2,500, 3,750, and 5,000 tokens of input. The number of LSTM units are 64, 128, and 256. These numbers are picked partially arbitrarily after some initial testing to find a suitably accurate model, which was using 15k tokens as the training size and 256 LSTM units.

There are a number of other configuration options that are left constant for all models. All models are trained for 50 epochs with a batch size of 16. The batch normalization layer has a momentum value of 0.75, and the dropout layer has a dropout rate of 0.15. The embedding layer has an embedding size of 16. The AMSGrad optimization algorithm is used with a learning rate of 0.0001 and a decay of 0.00001. The loss function is categorical cross-entropy. The testing set is acquired from a 90/10 percent split of the data.

To evaluate the models, I calculate perplexity, which can be calculated as 2 to the power of the categorical cross-entropy for the whole testing set.

Chapter 4

Results

First I look at the values of the reference metrics to see what to expect from the experimental results. Then I look at the part-of-speech tagging and language modeling tasks individually, and then compare them and develop the new metric of complexity.

4.1 Reference Metrics

The four referenced metrics, seen in Figure 4.1, show similar results with a few exceptions. Vocabulary size is the simplest to understand of the four. This is the only metric for which as data size increased, relative complexity increased. English appears to clearly be the least complex, and Chinese is the most complex, though Spanish, Japanese, and Chinese were relatively closer to each other than English. The standard deviation over the five calculations of the metric is very small at close to 1%.

For Type/Token Ratio, there is a similar graph to vocabulary size, with the main exception being that as data size increases, relative complexity decreases. English is the least complex,

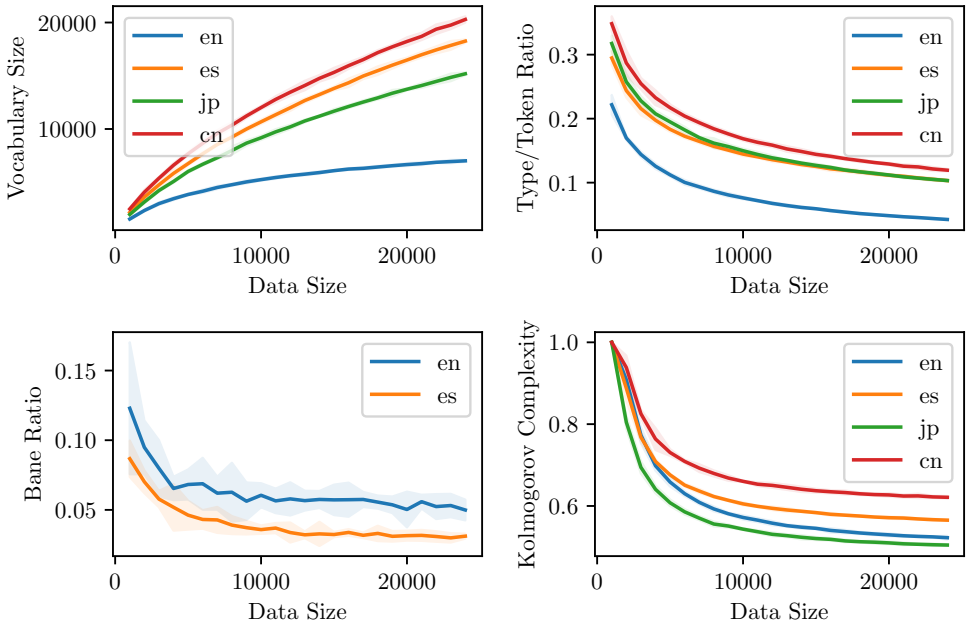

Figure 4.1: A plot comparing all languages for each of the reference metrics up to 24,000 tokens. In general, relative complexity tends to decrease as data size increases, with the exception of vocabulary size. Error margins are shown for three standard deviations around each point.

while Chinese is the most complex. In this case, Japanese is also slightly more complex than Spanish, though it almost drops below Spanish at larger data sizes.

For Bane Ratio, I only have data available for English and Spanish. Contrary to the other metrics, English scored higher in complexity for this metric. This may be because of English's many different types of affixes relative to number of stems compared to Spanish. While this disagrees with the other metrics, it is an interesting point to note that an information-theoretical metric gives this result. However, the standard deviation for calculating these results is unusually large, implying that this measure is not as reliable.

Finally for Kolmogorov complexity, there is a similar trend of decreasing complexity as data size increases, with Chinese being the most complex. However, here Japanese is the least complex, though English is not far off. In addition, English started as the second-most

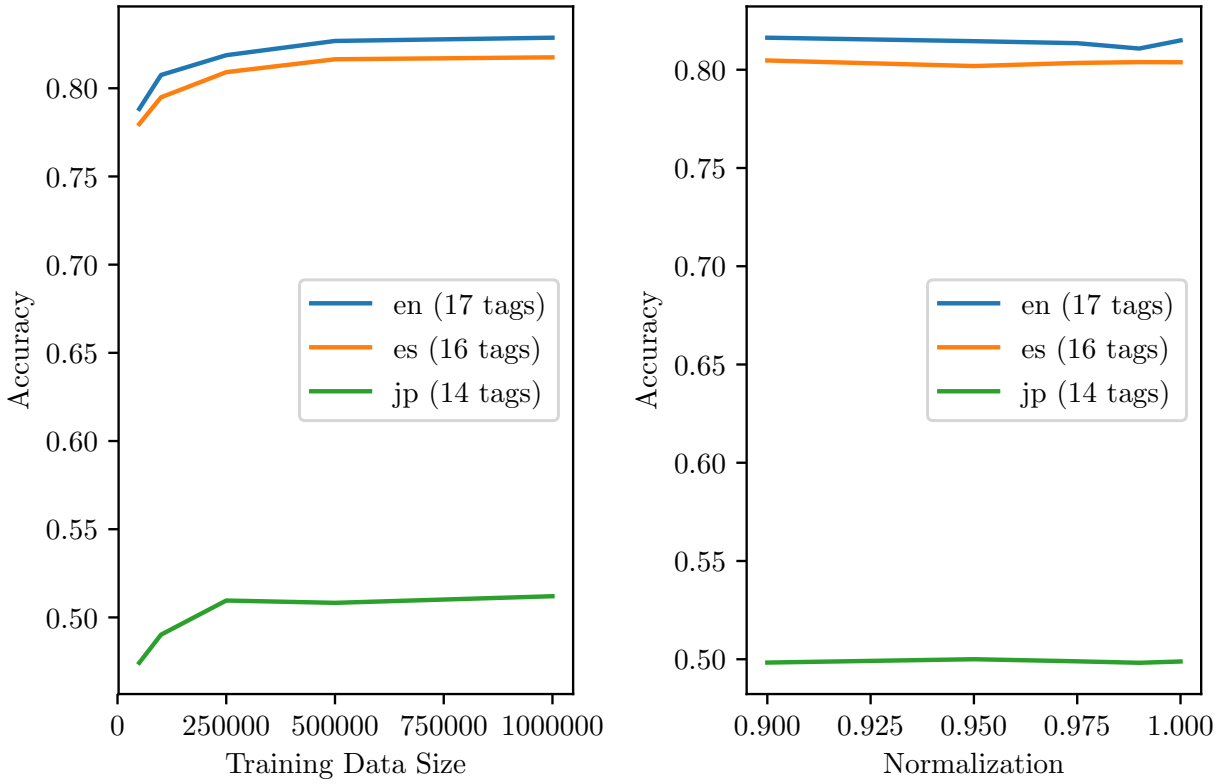

Figure 4.2: A plot with results from part-of-speech tagging experiments. Normalization seems to not have much effect, while training data size does to a certain extent.

complex, but then drops below Spanish. It may be that at low data sizes, English scores quite high but decreases more quickly than the other languages.

Overall, the main ideas that the reference metrics lead to are the following. First, English tends to be the least complex, while Chinese tends to be the most complex according to the metrics. Second, as data size increases, I expect a decrease in relative complexity, though it seems each value asymptotes over time to a more stable value at higher data sizes. In developing the complexity metric, I look to these points to ensure consistency with previous literature.

4.2 Part-of-Speech Tagging

The results of the part-of-speech tagging are displayed in Figure 4.2. The full data is available in Appendix A.2. For reference, there were 17 part-of-speech tags for English, 16 for Spanish, and 14 for Japanese. I vary two parameters for the logistic regression classifier, training data size and amount of normalization. Normalization appears not have very much effect on accuracy. When plotting training data size and normalization against accuracy on one graph, I also observe that as training data size increases, normalization also does not help much.

With accuracy against training data size, there is a slight curve that asymptotes as the training data size increases. Overall, English achieves the highest accuracy, and Japanese has the lowest by a wide margin. Spanish is slightly below English.

A higher accuracy indicates the model being more able to learn the language given the resources, which implies a lower morphological complexity. From the results, I would conclude here that English is the least complex language, while Japanese is considerably more complex in this context. This agrees with the reference metrics as I increase the size of training data.

4.3 Language Modeling

The results of the language modeling task are display in Figure 4.3. The full data are available in Appendix A.3. I vary two parameters for the LSTM language models as well, training data and number of LSTM units, while keeping all other parameters for the models constant.

The tests for both training size and LSTM units yield similar results. There appears to be an outlier in the case of Spanish, but because of the small number of data points, I include

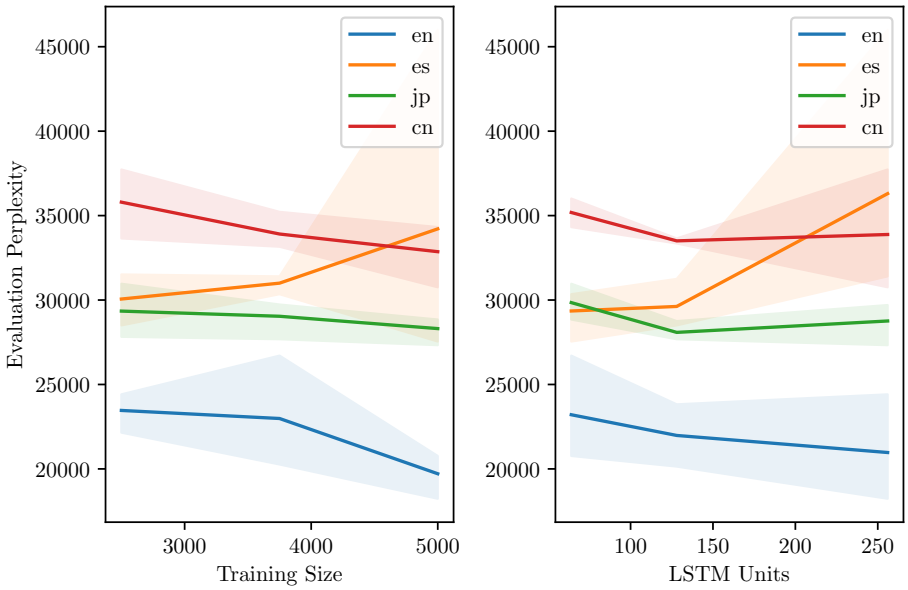

Figure 4.3: A plot with results from language modeling experiments. Varying both training data size and number of LSTM units seems to produce similar results. Each polygon captures the max and minimum values at each x axis value over all experiments.

it in the results. In general, evaluation perplexity decreases with increases in parameter complexity (larger training data sizes or more LSTM units).

Low perplexities indicate better performance, so English performed the best and therefore implies least morphological complexity. Japanese and Spanish are quite close, similar to what was observed with vocabulary size and type/token ratio. And generally, Chinese was considered the most complex. These agree with the reference metrics as I increase the size of training data.

Unlike with the part-of-speech tagging tasks, it is difficult to tell if perplexity will asymptote from the results alone. However, I expect perplexity to asymptote as training data size increases. It is unclear if the currently observed order of languages will change without further experimentation.

4.4 Comparison and the Metric

Overall, the evaluation results from each of the machine learning tasks, whether it be using accuracy for part-of-speech tagging or perplexity for language modeling, are proportional to the results from the reference metrics. English has been consistently the least complex, meaning it had the highest accuracy and lowest perplexity on the various tasks. Chinese is the most complex in the language modeling tasks, and Japanese was the most complex in the part-of-speech tagging tasks.

Our evaluation metrics also displayed the same asymptotic behavior as the reference metrics. More importantly than the fact that they asymptote, all languages appear to asymptote at the same rate as each other, staying in relative order. Therefore even as the values change with differing amounts of training data or different model configurations, the languages will likely stay in the same order of complexity.

Knowing this, I propose these model evaluation metrics as a relative metric of morphological complexity. The metric to use specifically will depend on the machine learning task and model. For the tests I used the standard evaluation metrics for the respective tasks.

Using this metric, the following procedure can be used to compare morphological complexities between languages. If labeled data is available, a task like part-of-speech tagging can be used even with a simple algorithm like logistic regression. The same model configuration should be used to train identical models using two datasets containing different languages. Given the same amount of training data and the same model configuration, the model with the higher accuracy indicates the less complex language. If labeled data is unavailable, a task like language modeling can be used. Again, the same model configurations for LSTM language models can be used to train identical models with different language datasets. The model with the lower perplexity is the less complex language.

Chapter 5

Discussion

In this thesis I propose using machine learning evaluation metrics such as accuracy and perplexity as relative morphological complexity metrics for different languages. I test using part-of-speech tagging tasks with a logistic regression classifier and language modeling tasks with an LSTM-based neural network with the languages English, Spanish, Japanese, and Chinese. For each task, I keep all configuration constant while varying one configuration item for each language to observe how the model's evaluation changes. I then compare these results between different languages to observe how the same models with same configurations differ when trained on different language data. I compare these results to four reference metrics from linguistics literature and conclude that accuracy and perplexity behave similarly to those reference metrics, meaning I can conclude an ordering of language complexities using those metrics.

By using machine learning models to learn language without much pre-processing or adjusting models to fit specific languages, this approach provides an unbiased estimator of morphological complexity for languages relative to each other. This does not provide an

absolute measure of morphological complexity for a single language, but does allow for comparison between languages.

Using machine learning model evaluation metrics as the metric for relative morphological complexity suffers from only two potential sources of bias. The first is the training data used for each model. As with any machine learning tasks, having unbiased training data is necessary for developing accurate and useful models. Though this approach aims to be as simple as possible to reduce the need for perfectly pre-processed training data, the training data should be as representative of its language as possible.

The second potential source of bias comes from the tools used to create and evaluate the machine learning models. In my tests, I experienced trouble with Bane's Linguistica library due to its reliance on Latin character sets. In addition, English was consistently noted as the least complex language. It is possible this is due to the tokenizers and libraries used when processing input for my models being tuned specifically for English. Again, my approach aims to be as simple as possible to reduce this bias when possible, such as by doing no pre-processing or feature extraction.

5.1 Type of Complexity

Morphology refers to the study of words in a language and how they relate to other words, primarily through their constituent parts such as stems and affixes. The methods in this thesis do explore the relationships between words of a language and how the complexity of the vocabulary of a language, as reflected by a representative corpus, relates to other languages. However, because of the lack of explicit data on individual morphemes in each language, the experiments do not strictly capture morphological complexity. This then begs the question of what kind of complexity is captured.

One potential option is a sort of "word complexity," or the complexity of unique words in the vocabulary. Without needing more detailed information on the words, the machine learning models are able to learn semantic relationships between words by using word embeddings that were used in the LSTM model. By approaching languages from an information-theoretic perspective, an idea of how much information is being conveyed per language is also established, which can be used to compare how concise different languages are. This information can be determined by looking at words in their sentence context, without needing other morphological detail. "Word complexity" describes this by focusing on the whole words themselves along with their context.

Another option is a "vocabulary complexity" or a "descriptive complexity." Because the size of the vocabulary, or unique tokens, in the language seemed to have a large effect on its apparent complexity, the models may be learning specifically about the variety in the vocabulary and how it is used in sentences, without as much focus on how the vocabulary is constructed, which would refer more to word morphology and how words are put together. By this definition, languages with a larger variety of words to describe the world would appear more complex, while languages that employ more assumptions using context and don't directly describe the world would appear less complex. This may explain why Japanese appeared least complex in the LSTM tasks and in Kolmogorov complexity, since Japanese speakers and writers tends to drop many assumed words from sentences. On the contrary, Chinese, which has many different characters that may describe similar concepts would appear more complex.

5.2 Future Work

There are a few ways this thesis can be expanded and improved. First, the experiments performed used small subsets of the whole training dataset available in order to facilitate the

large number of experiments. Each experiment involved training a full model which requires time and computational resources. Given more time and resources, more models could be trained with larger training datasets to evaluate how the models perform at the tails of their asymptotic behavior.

Second, experiments could be performed by varying more configuration items for the models. There were many configuration items for the neural networks specifically such as embedding size and batch size that could be varied and observed to see if the same behavior holds.

Third, experiments could be performed using more complex models. For example, using neural networks built using Transformer architectures rather than LSTMs, or using hybrid models similar to state-of-the-art models. This would require more computational resources to train, but more complex may show different behavior or asymptote at a different rate.

Finally, this could be applied the many more languages. Four languages provided a decent baseline of consistency, but by analyzing more languages from different backgrounds and language families we could verify the results hold for many other languages.

The experimental methods of the thesis could also be fundamentally changed to explore other aspects of what machine learning models learn about language. Repeating the experiments with character-level models as opposed to word-level models may be able to actually parse morphological structure from words, providing a better measure of morphological complexity. Alternatively, preprocessing the words through an embedding model before calculating reference metrics and being input to models could provide different results based more on the semantics of the words.

Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] M. Bane. Quantifying and measuring morphological complexity. 2007.

[3] C. Bentz, T. Ruzsics, A. Koplenig, and T. Samardisamardic. A comparison between morphological complexity measures: Typological data vs. language corpora. 12 2016.

[4] F. Chollet et al. Keras. `https://keras.io`, 2015.

[5] R. Cotterell, S. J. Mielke, J. Eisner, and B. Roark. Are all languages equally hard to language-model? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 536–541, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[6] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier. Language modeling with gated convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 933–941. JMLR.org, 2017.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2018.

[8] S. Edunov, M. Ott, M. Auli, and D. Grangier. Understanding back-translation at scale. In *EMNLP*, 2018.

[9] K. Ehret. Kolmogorov complexity as a universal measure of language complexity. In A. Berdicevskis and C. Bentz, editors, *Proceedings of the First Shared Task on Measuring Language Complexity*, pages 8–14, 2018.

[10] C. Gong, D. He, X. Tan, T. Qin, L. Wang, and T.-Y. Liu. Frage: Frequency-agnostic word representation. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-

Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 1334–1345. Curran Associates, Inc., 2018.

[11] X. Gutierrez-Vasques and V. Mijangos. Comparing morphological complexity of Spanish, Otomi and Nahuatl. In *Proceedings of the Workshop on Linguistic Complexity and Natural Language Processing*, pages 30–37, Santa Fe, New-Mexico, Aug. 2018. Association for Computational Linguistics.

[12] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[13] P. Juola. Measuring linguistic complexity: The morphological tier. *Journal of Quantitative Linguistics*, 5(3):206–213, 1998.

[14] D. Kimura and K. Tanaka-Ishii. A study on constants of natural language texts. *Journal of Natural Language Processing*, 21:877–895, 01 2014.

[15] J. L. Lee and J. A. Goldsmith. Linguistica 5: Unsupervised learning of linguistic structure. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 22–26, San Diego, California, June 2016. Association for Computational Linguistics.

[16] M. Li and P. Vitnyi. *An Introduction to Kolmogorov Complexity and Its Applications*. 01 1997.

[17] P. Lison and J. Tiedemann. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*, 2016.

[18] H. Liu. Dependency distance as a metric of language comprehension difficulty. *Journal of Cognitive Science*, 9:159–191, 09 2008.

[19] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur. Recurrent neural network based language model. In *INTERSPEECH*, 2010.

[20] F. J. Newmeyer and L. B. Preston, editors. *Measuring Grammatical Complexity*. Oxford University Press, 2014.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[22] T. Pranckeviius and V. Marcinkeviius. Application of logistic regression with part-of-the-speech tagging for multi-class text classification. In *2016 IEEE 4th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE)*, pages 1–5, Nov 2016.

[23] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[24] B. Sagot. Comparing Complexity Measures. In *Computational approaches to morphological complexity*, Paris, France, Feb. 2013. Surrey Morphology Group.

[25] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423, 1948.

[26] S. Takahashi and K. Tanaka-Ishii. Do neural nets learn statistical laws behind natural language? *PLOS ONE*, 12(12):1–17, 12 2017.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.

[28] K. von Prince and V. Demberg. Pos tag perplexity as a measure of syntactic complexity. In A. Berdicevskis and C. Bentz, editors, *Proceedings of the First Shared Task on Measuring Language Complexity*, pages 20–25, 2018.

Appendix A

Full Experiment Results

A.1 Reference Metric Results

A.1.1 English

Table A.1: Results for each of the reference metrics for English for LSTM model data sizes.

Data Size	Vocab Size	T/T Ratio	Bane Ratio	Kolmogorov
2,500	2673	0.1551	0.08359	0.8388
3,750	3328	0.1283	0.07350	0.7134
5,000	3822	0.1111	0.06876	0.6551

A.1.2 Spanish

Table A.2: Results for each of the reference metrics for Spanish for LSTM model data sizes.

Data Size	Vocab Size	T/T Ratio	Bane Ratio	Kolmogorov
2,500	4196	0.2276	0.06330	0.8142
3,750	5608	0.2023	0.05358	0.7241
5,000	6777	0.1827	0.04627	0.6759

A.1.3 Japanese

Table A.3: Results for each of the reference metrics for Japanese for LSTM model data sizes.

Data Size	Vocab Size	T/T Ratio	Bane Ratio	Kolmogorov
2,500	3730	0.2434	n/a	0.7411
3,750	4954	0.2150	n/a	0.6526
5,000	5998	0.1943	n/a	0.6091

A.1.4 Chinese

Table A.4: Results for each of the reference metrics for Chinese for LSTM model data sizes.

Data Size	Vocab Size	T/T Ratio	Bane Ratio	Kolmogorov
2,500	4692	0.2683	n/a	0.8617
3,750	6252	0.2352	n/a	0.7738
5,000	7717	0.2175	n/a	0.7307

A.2 Part-of-Speech Tagging Results

A.2.1 English

Table A.5: Results for each of the part-of-speech tagging tasks for English.

Tokens	Normalization	Accuracy	Unique Tokens
50000	1	0.797954887	4057
50000	0.99	0.790676692	4067
50000	0.975	0.785503759	4050
50000	0.95	0.777263158	4037
50000	0.9	0.790075188	4060
100000	1	0.808992481	5134
100000	0.99	0.810977444	5087
100000	0.975	0.807909774	5125
100000	0.95	0.803879699	5168
100000	0.9	0.805894737	5076
250000	1	0.818875188	6515
250000	0.99	0.817251128	6488
250000	0.975	0.819067669	6497
250000	0.95	0.820258647	6515
250000	0.9	0.818273684	6486
500000	1	0.826508271	7434
500000	0.99	0.825780451	7464
500000	0.975	0.827615038	7435
500000	0.95	0.826261654	7438
500000	0.9	0.82795188	7471
1000000	1	0.8291759	8022
1000000	0.99	0.828021129	8022
1000000	0.975	0.827447645	8022
1000000	0.95	0.826257763	8022
1000000	0.9	0.832437346	8022

A.2.2 Spanish

Table A.6: Results for each of the part-of-speech tagging tasks for Spanish.

Tokens	Normalization	Accuracy	Unique Tokens
50000	1	0.783819549	7147
50000	0.99	0.77924812	7269
50000	0.975	0.778165414	7213
50000	0.95	0.780330827	7243
50000	0.9	0.777082707	7148
100000	1	0.797533835	10854
100000	0.99	0.794616541	10911
100000	0.975	0.794315789	10818
100000	0.95	0.793894737	10908
100000	0.9	0.793984962	10891
250000	1	0.808890226	18119
250000	0.99	0.805906767	18031
250000	0.975	0.809106767	18091
250000	0.95	0.8096	17994
250000	0.9	0.812162406	17959
500000	1	0.816126316	25435
500000	0.99	0.816330827	25505
500000	0.975	0.816757895	25334
500000	0.95	0.816252632	25460
500000	0.9	0.816757895	25380
1000000	1	0.817196992	34982
1000000	0.99	0.81321203	34848
1000000	0.975	0.818965414	34721
1000000	0.95	0.819287218	34679
1000000	0.9	0.819031579	34930

A.2.3 Japanese

Table A.7: Results for each of the part-of-speech tagging tasks for Japanese.

Tokens	Normalization	Accuracy	Unique Tokens
50000	1	0.469473684	9577
50000	0.99	0.475007519	9404
50000	0.975	0.471097744	9535
50000	0.95	0.477233083	9509
50000	0.9	0.478556391	9575
100000	1	0.490285714	14649
100000	0.99	0.492992481	14453
100000	0.975	0.491488722	14554
100000	0.95	0.488661654	14597
100000	0.9	0.487578947	14640
250000	1	0.509521805	24481
250000	0.99	0.511181955	24413
250000	0.975	0.51041203	24598
250000	0.95	0.509654135	24498
250000	0.9	0.507115789	24687
500000	1	0.507590977	35368
500000	0.99	0.510153383	35435
500000	0.975	0.509630075	35442
500000	0.95	0.505100752	35370
500000	0.9	0.508806015	35361
1000000	1	0.514673684	49549
1000000	0.99	0.510715789	49506
1000000	0.975	0.512105263	49625
1000000	0.95	0.510330827	49689
1000000	0.9	0.512219549	49697

A.3 Language Modeling Results

A.3.1 English

Table A.8: Results for each of the language modeling tasks for English.

Training Size	LSTM Units	Max Sequence Length	Total Words	Eval Loss	Eval Perplexity
2500	64	58	8536	8.943550437	22149.29157
2500	128	58	8536	9.157530013	23823.73143
2500	256	58	8536	9.350491445	24414.65583
3750	64	58	8536	9.526291554	26708.7171
3750	128	58	8536	8.981067291	21981.89799
3750	256	58	8536	8.367787104	20263.76149
5000	64	58	8536	8.703749201	20767.34192
5000	128	58	8536	8.776796776	20138.86999
5000	256	58	8536	8.453456764	18234.53458

A.3.2 Spanish

Table A.9: Results for each of the language modeling tasks for Spanish.

Training Size	LSTM Units	Max Sequence Length	Total Words	Eval Loss	Eval Perplexity
2500	64	82	83860	10.88528007	30165.67415
2500	128	82	83860	10.59416642	28497.23959
2500	256	82	83860	10.932513	31514.66381
3750	64	82	83860	10.65911132	30344.93239
3750	128	82	83860	10.91572923	31251.98543
3750	256	82	83860	10.36546736	31402.03576
5000	64	82	83860	10.3025796	27542.39688
5000	128	82	83860	10.3567289	29111.09719
5000	256	82	83860	12.84245551	45989.01037

A.3.3 Japanese

Table A.10: Results for each of the language modeling tasks for Japanese.

Training Size	LSTM Units	Max Sequence Length	Total Words	Eval Loss	Eval Perplexity
2500	64	39	30212	10.7696001	30964.67443
2500	128	39	30212	10.27066716	27821.64707
2500	256	39	30212	10.39982494	29247.06246
3750	64	39	30212	10.32876348	29733.61599
3750	128	39	30212	10.42400536	27678.6543
3750	256	39	30212	10.58932414	29703.23227
5000	64	39	30212	10.08694808	28846.46524
5000	128	39	30212	10.29054392	28752.30553
5000	256	39	30212	10.31835362	27326.88227

A.3.4 Chinese

Table A.11: Results for each of the language modeling tasks for Chinese.

Training Size	LSTM Units	Max Sequence Length	Total Words	Eval Loss	Eval Perplexity
2500	64	99	56292	11.48050089	36010.74105
2500	128	99	56292	11.09059214	33646.68778
2500	256	99	56292	11.67941991	37729.77352
3750	64	99	56292	10.98018001	35222.22386
3750	128	99	56292	10.79259286	33349.29838
3750	256	99	56292	11.1680774	33156.0655
5000	64	99	56292	10.94011411	34319.13986
5000	128	99	56292	10.73741313	33512.44656
5000	256	99	56292	10.6963689	30755.4809